8. Nicolas

Program Name: Nicolas.java **Input File: nicolas.dat**

It's February, which is when Texas gets the most snow, which is no different for Nicolas on this cold morning. Thankfully, Nicolas lives within walking distance of his school; however, unfortunately for him, the roads and sidewalks are rather icy. This makes it so that, when Nicolas starts walking in a certain direction, he is forced to continue moving in that direction until he bumps into something stationary. Thankfully for him, Nicolas is rather tough and is not concerned with getting hurt from running into said objects; however, he is concerned with minimizing the number of times he bonks into something, even if it means walking (really sliding on the ice) a little further. Help Nicolas in planning the path that minimizes the number of times he needs to run into a stationary object on his way to school.

Input: The first line of input will consist of a single integer n, $1 \leq n \leq 10^3$, denoting the number of different places that Nicolas needs to determine a route to. The next n test cases will each consist of a single line denoting two space separated integers, r and c, $2 \leq r, c \leq 2 \cdot 10^2$, denoting the number of rows and columns, respectively, in the map that Nicolas is basing his routes off of. The next r lines will consist of c characters, each character will be among one of the following:

- `'.'` : Denotes a walkable space that Nicolas can slide on.
- `'#'` : Denotes a stationary object that Nicolas can bump into.
- `'S'` : Denotes Nicolas' starting position. There will be a single spot labeled with an `'S'`.
- `'E'` : Denotes the location that Nicolas is attempting to reach. There will be a single spot labeled with an `'E'`.

Output: For each of Nicolas' n routes, print out a string denoting the series of directions that Nicolas should move in to minimize the number of stationary objects that he has to run into. This string will consist of the characters `'N'`, `'S'`, `'E'`, and `'W'` denoting North, South, East, and West. If there are multiple such paths that minimize this, then print out the one that is lexicographically smallest. It is also guaranteed that there will be some path from where Nicolas is starting and where he intends to end at. You may also assume that there is an understood border of stationary objects on the spaces outside of the map provided to you.

Sample input:
```
3
5 6
S###.E
.#.#.#
.#.#.#
......
#..#..
5 10
S.........
..#######.
#..######.
##..#####.
###..E....
3 5
..S..
.###.
..E..
```

Sample output:
```
SESWNE
ESW
ESW
```